Are Cache Attacks on Public Clouds Practical?

Thomas Eisenbarth
Joint work with Gorka Irazoqui, Mehmet Sinan Inci, Berk Gulmezoglu and Berk Sunar

WPI - 10/19/2015
Outline

• Cloud Computing and Isolation
• Extracting Information from Co-located VM
• Attacking AES across VM Boundaries
• A Practical RSA Key Recovery
Cloud Computing

- Computation increasingly outsourced to cloud servers
- CSPs: many users on shared, homogeneous platforms
- Users rent VMs, share same computer
- Shared resources ⇒ Information Leakage?
Security through Isolation

• Virtual machines: Abstraction of physical machine
• Hypervisor (VMM) ensures Isolation through virtualization
• VMs might *feel* each other’s load on some low-level resources → potential side channels
Outline

• Cloud Computing and Isolation
• **Extracting Information from Co-located VM**
• Attacking AES across VM Boundaries
• A Practical RSA Key Recovery
Cross-VM Side Channel Attack

Suitable covert channel in the cloud?

– Cross Core: Last Level Cache (L3 Cache) accesses
  ➔ Adversary and victim share full access to L3 cache
  ➔ Cache Access cannot be virtualized (70x slowdown)
How to track victim’s data?

**Deduplication**

- Keeps only one copy of duplicate data in RAM

- When Target VM accesses page
  - page copied to cache: copy in shared LLC
  - Subsequent Spy VM access also faster!
  - Spy can detect Target VMs accesses to known pages
Flush+Reload Attack: Concept

Steps:

1. **Flush** desired memory lines
2. Wait for some time
3. **Reload** memory lines and measure reload time.

**Clean detection if monitored memory line was accessed**
Outline

• Cloud Computing and Isolation
• Extracting Information from Co-located VM
• **Attacking AES across VM Boundaries**
• A Practical RSA Key Recovery
Target Cipher: AES

AES T-table implementation:

- T-tables stored in memory/cache

Idea:
- Detect T-table accesses in last round
- Ciphertext mapping to monitored table position is always accessed
- Inclusive caches ensure T-table in LLC
Attack Setup and Results

Test setup

- Intel i5-3320
- OpenSSL 1.0.1f AES C implementation
- Ubuntu 12.04 OS
- VMware ESXI 5.5
- Transparent Page Sharing enabled (NO LONGER default)

Measurements take less than a minute!


Are Cross-VM Cache Attacks Realistic?

Cross-VM Flush+Reload Attack on AES works if

- Server has a shared level of cache ✅
- Attacker and the victim are physically co-located ❓
- VMM implements memory deduplication ✖️

- Memory Deduplication can enable Cross-VM cache attacks
  - [http://kb.vmware.com/kb/2080735](http://kb.vmware.com/kb/2080735)
Cache Attacks without Deduplication?

• Cache attacks are old [Hu92]
• General technique: *Prime+Probe* [OST06]:

  1. **Flush Prime** desired memory lines  
      *fill monitored cache lines with data making an eviction set*
  2. Wait for some time
  3. **Reload Probe** memory lines  
      *read eviction set data and time read*

• Problems:
  – Usually only applied on L1-Cache → not cross-core
  – L3-Cache is too large (25MB vs 64kB) and cannot be controlled by spy

Classic: Prime+ Probe in L1 Cache

- **Eviction Set** fills one cache set (dummy data)
- Intel: L1$ is virtually indexed and physically tagged → attacker controls/knows set number → Eviction set is easily constructed

Not true for LLC: set index is part of virtual address
Prime + Probe in LLC

How to gain control over LLC?

– Huge memory pages! 2MB pages instead of 4 KB.
– Offset becomes 21 bits

→ Eviction set for L3$ can be constructed
LLC Prime+Probe on AES

- Same target as before (Single line of T-Table)

- **Preparation:** Need to locate T-Table in LLC

**TestSetup 1:**
- Intel I5-650
- 2 cores
- Xen and Vmware + Ubuntu 12.04

**TestSetup 2:**
- Intel e5-2640
- 8 cores
- Vmware + Ubuntu 14.04


Are Cross-VM Cache Attacks Realistic?

Cross-VM Flush+Reload Attack on AES works if

• Server has a shared level of cache

• Attacker and the victim are physically co-located

• VMM implements memory deduplication
Outline

• Cloud Computing and Isolation
• Extracting Information from Co-located VM
• Attacking AES across VM Boundaries
• AES Attack #2: Without Deduplication
• A Practical RSA Key Recovery
Co-location

First (and last) success in 2009 [RTS09]:
1. Launch many instances on cloud
2. Check if any are co-located

- How to detect Co-location?
  - Ping time?
  - IP address of instance or hypervisor?
  - Disk Load?

Test Setup

- AWS EC2 m2.medium instances:
  - Intel Xeon E5 2670 v2 CPU @ 2.5 GHz
  - 10 cores share 25 MB of L3 cache
  - Modified (Hardened) Xen VMM
  - Up to 10 co-located instances (VMs)

- 4 accounts w/ 20 instances (no within-acc colocation)
- Ping is constant time
- HDDs replaced with SSDs
- Dom0 IPs hidden

New Co-location detection needed
Co-Location Attempt:
LLC Cache Accesses

+ Works reliable and we know how to do it
+ Impossible* to block
- Requires slice recovery
- Noise?

• Gives Reliable Co-location Detection
• ensures that cache attack will work

Alternative: Memory bus contention [XWW15,VZRS15]

[XWW15] XU, Z., WANG, H., AND WU, Z. A measurement study on co-residence threat inside the cloud. USENIX Security 15
Target Cryptosystem

• Libgcrypt 1.6.2 ‘s RSA implementation
  – RSA CRT with 2048 bit modulus size
  – Sliding window exponentiation (5 bits)
  – Message blinding to prevent chosen ciphertext attacks

Is this state-of-the-art?

• Libgcrypt 1.6.3 (February 2015)
  – Table accesses now constant execution flow
    (no more cache games)
Attack on RSA-CRT Sliding Window

1. Find cache trace of sliding window multiplicands
2. Observe several exponentiations to reduce noise
3. Align observations to reduce noise
4. Run error correcting key recovery to fix errors introduced by noise
Identifying a Correct Cache Line

- 10x2048 cache lines
- Source code reveals approximate position
- Search through remaining choices
- Once found, repeat observations
Raw Traces
After Alignment
After Processing and Alignment

• Correct (red) vs recovered (blue):
  → little remaining noise
Final key recovery?

• Distance to table initialization reveals multiplicand value
• $d$ must be recovered from noisy $d_p$ and $d_q$

More details in:
http://eprint.iacr.org/2015/898
Conclusion

• Co-Location Problem can be solved in Public Clouds
• Caches provide a powerful side channel in the Cloud
• Deduplication makes exploitation very simple
• Smart Prime+Probe works w/out Deduplication
→ Key Recovery in Public Clouds is possible!

– Countermeasures still open problem:
  Many proposed, but cost overhead prohibitive?
• For Crypto Libraries:
  Recent patches of well-maintained libraries are secure
Thank you!

vernam.wpi.edu

teisenbarth@wpi.edu